WHAT IS CLAIMED IS:

1. A method of building a compressed lexicon, comprising:

 receiving a word list and word-dependent data associated with each word in the word list;

 selecting a word from the word list;

 generating an index entry identifying a location in a lexicon memory for holding the selected word;

 encoding the selected word and its associated word-dependent data to obtain encoded words and associated encoded word-dependent data; and

 writing the encoded word and its associated word-dependent data at the identified location in the lexicon memory.

2. The method of claim 1 and further comprising:

 repeating the steps of selecting, generating, encoding and writing for each word in the word list and the associated word-dependent data.

3. The method of claim 2 and further comprising:

 writing codebooks corresponding to the encoded words and the encoded word-dependent data in the lexicon memory.

4.   The method of claim 1 wherein receiving the word
list comprises:

    counting the words in the word list;

    allocating a hash table memory based on a number
        of words in the word list; and

    allocating a lexicon memory based on the number
        of words in the word list.

5.   The method of claim 1 wherein generating an
index entry comprises:

    determining a next available location in the
        lexicon memory.

6.   The method of claim 5 wherein generating an
index entry comprises:

    calculating a hash value for the selected word;

    indexing into the hash table to an index
        location based on the hash value; and

    writing location data identifying the next
        available location in the lexicon memory
        into the index location in the hash table.

7.   The method of claim 6 wherein writing location
data comprises:

    writing an offset into the lexicon memory that
        corresponds to the next available location
        in the lexicon memory.

8.   The method of claim 1 wherein encoding
comprises:

providing a word encoder to encode the words in
the word list and encoding the words with
the word encoder; and

providing word-dependent data encoders for each
type of word-dependent data in the word
list and encoding the word-dependent data
with the word-dependent data encoders.

9.   The method of claim 8 wherein encoding further
comprises:

Hufmann encoding the selected word and its
associated word-dependent data.

10.   The method of claim 1 wherein writing the
encoded word and word-dependent data comprises:

writing a data structure comprising:

a word portion containing the encoded word;

a word-dependent data portion containing
the encoded word-dependent data; and

wherein each word-dependent data portion
has an associated last indicator
portion and word-dependent data
indicator portion, the last indicator
portion containing an indication of a
last portion of word-dependent data
associated with the selected word, and
the word-dependent data indicator
portion containing an indication of
the type of word-dependent data stored

in the associated word dependent data portion.

11.   The method of claim 10 wherein writing a data structure comprises writing the word portion and the word-dependent data portions as variable length portions followed by a separator.

12.   A method of accessing word information related to a word stored in a compressed lexicon, comprising:
    receiving the word;
    accessing an index to obtain a word location in
        the compressed lexicon that contains
        information associated with the received
        word;
    reading encoded word information from the word
        location; and
    decoding the word information.

13.   The method of claim 12 and further comprising:
    prior to reading the encoded word information,
        reading an encoded word from the word
        location;
    decoding the encoded word; and
    verifying that the decoded word is the same as
        the received word.

14.   The method of claim 12 wherein reading the encoded word information comprises:

reading a plurality of fields from the word
location containing variable length word
information.

15. The method of claim 14 wherein reading a
plurality of fields comprises:
prior to reading each field, reading data type
header information indicating a type of
word information in an associated field.

16. The method of claim 15 wherein reading a
plurality of fields comprises:
reading a last field indicator indicating
whether an associated one of the plurality
of fields is a last field associated with
the received word.

17. The method of claim 12 wherein decoding the word
information comprises:
initializing decoders associated with the word
and its associated information.

18. The method of claim 12 wherein accessing an
index comprises:
calculating a hash value based on the received
word;
finding an index location in the index based on
the hash value; and

reading from the index location a pointer value pointing to the word location in the compressed lexicon.

19. A compressed lexicon builder for building a compressed lexicon based on a word list containing a plurality of domains, the domains including words and word-dependent data associated with the words, the compressed lexicon builder comprising:

a plurality of domain encoders, one domain encoder being associated with each domain in the word list, the domain encoders being configured to compress the words and word-dependent data to obtain compressed words and compressed word-dependent data;

a hashing component configured to generate a hash value for each word in the word list;

a hash table generator, coupled to the hashing component, configured to determine a next available location in a lexicon memory and write, at an address in a hash table identified by the hash value, the next available location in the lexicon memory; and

a lexicon memory generator, coupled to the domain encoders and the hash table generator, configured to store in the lexicon memory the compressed words and compressed word-dependent data, each compressed word and its associated

compressed word-dependent data being stored
at the next available location in the
lexicon memory written in the hash table at
the hash table address associated with the
compressed word.

20. The compressed lexicon builder of claim 19
wherein the lexicon memory generator is configured to
store the compressed words and associated compressed
word-dependent data in variable length word fields
and variable length word-dependent data fields in the
lexicon memory.

21. The compressed lexicon builder of claim 20
wherein the lexicon memory generator is configured to
store header information associated with each word-
dependent data field indicating whether the word-
dependent data field is a last field associated with
the compressed word and indicating a type of word-
dependent data stored in the word-dependent data
field.

22. The compressed lexicon builder of claim 19 and
further comprising:
    a codebook generator generating a codebook
        associated with each domain encoder.

23. A compressed lexicon accesser for accessing
word-dependent data in a compressed lexicon based on
a received word, the compressed lexicon accesser
comprising:

a plurality of domain decoders, one domain
decoder being associated with each domain
in the compressed lexicon, the domain
decoders being configured to decompress the
words and word-dependent data;

a hashing component configured to generate a
hash value for the received word;

a hash table accesser, coupled to the hashing
component, configured to read from an
address in a hash table identified by the
hash value, a word location in a lexicon
memory corresponding to a lexicon entry for
the received word; and

a lexicon memory accesser, coupled to the domain
decoders and the hash table accesser,
configured to read from the word location
in the lexicon memory compressed words and
compressed word-dependent data and provide
the compressed words and compressed word-
dependent data to corresponding domain
decoders.

24. The compressed lexicon of claim 23 wherein the
lexicon memory accesser is configured to read the
compressed words and associated compressed word-
dependent data from variable length word fields and

variable length word-dependent data fields in the lexicon memory.

25. The compressed lexicon of claim 24 wherein the lexicon memory accesser is configured to read header information associated with each word-dependent data field indicating whether the word-dependent data field is a last field associated with the compressed word and indicating a type of word-dependent data stored in the word-dependent data field.

26. The compressed lexicon of claim 23 and further comprising:

  a codebook accesser accessing a codebook

    associated with each domain decoder.

27. A compressed lexicon having a data structure, comprising:

  a word portion storing a compressed word;

  a first word-dependent data portion storing a

    first type of compressed word-dependent

    data; and

  a first header portion associated with the first

    word-dependent data portion storing a type

    indicator indicating the type of word-

    dependent data stored in the first word-

    dependent data portion, and a last field

    indicator indicating whether the first

    word-dependent data portion is a last word-

dependent data portion associated with the compressed word.

28. The compressed lexicon of claim 27 wherein the data structure comprises:

    a plurality of word portions;

    a plurality of word-dependent data portions associated with each word portion; and

    a plurality of header portions, one header portion being associated with each word-dependent data portion.

29. The compressed lexicon of claim 27 and further comprising:

    a plurality of marker portions each marker portion marking an end of each word portion or a word-dependent data portion.

30. The compressed lexicon of claim 27 and further comprising:

    a codebook portion storing a plurality of codebooks, one codebook being associated with the word portion and each type of word-dependent data portion.

31. The compressed lexicon of claim 27 and further comprising:

    an index having a pointer to the word portion, wherein the pointer is stored at an address in the index identified by a hash value

associated with the word compressed in the
word portion.